

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-233326

(43)Date of publication of application : 10.09.1993

(51)Int.Cl.

G06F 9/46

(21)Application number : 04-295194

(71)Applicant : INTERNATL BUSINESS MACH  
CORP <IBM>

(22)Date of filing : 04.11.1992

(72)Inventor : CARNEY WILLIAM PETER  
ENGLAND LAURENCE EDWARD  
HOCHMUTH GARY JOHN  
OWINGS BRIAN  
PORTER ERIC LYNN  
SHANNON ALFRED WILLIAM  
WILSON ROBERT AARON

(30)Priority

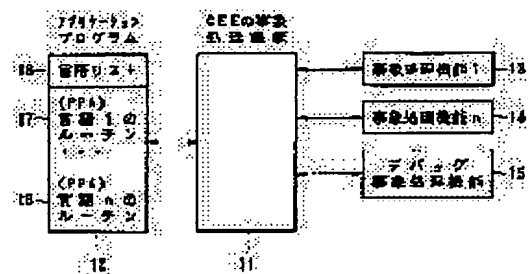
Priority number : 91 810619 Priority date : 19.12.1991 Priority country : US

## (54) METHOD AND SYSTEM FOR HANDLING EVENT IN COMPUTER SYSTEM

(57)Abstract:

PURPOSE: To provide a method and system for handling an event with a related parameter at the time of executing a program.

CONSTITUTION: In order to handle the events within a computer system, which is generated in the middle of executing the program 12 including routines in plural programming languages, the number and identification of respectively unique program languages are decided by a language list 16 or another equivalent means. Unique event handling means (event processing functions) 13 to 15 are initialized by each unique programming language and at the time of executing the program, the selected event of the event processing function is detected and its parameter is decided. The detected event is divided into a multi-address communication or desired events, a multi-address communication events (except for a debugging event processing function) are sent to all the event processing functions 13 and 14, and a desired event is event sent to one event processing function. An event code and the related parameter are sent to the event processing functions 13 to 15. These event processing functions 13 to 15 generate a proper returning record supporting the success, failure, not-processing of the event and returns necessary information to each selected event.



LEGAL STATUS

[Date of request for examination] 04.11.1992  
[Date of sending the examiner's decision of rejection] 16.09.1997  
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]  
[Date of final disposal for application]  
[Patent number]  
[Date of registration]  
[Number of appeal against examiner's decision of rejection]  
[Date of requesting appeal against examiner's decision of rejection]  
[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19)日本国特許庁(JP)

(12)公開特許公報(A)

(11)特許出願公開番号

特開平5-233326

(43)公開日 平成5年(1993)9月10日

(51)Int.Cl.<sup>5</sup>

G 0 6 F 9/46

識別記号

3 4 0 A 8120-5B

庁内整理番号

F I

技術表示箇所

審査請求 有 請求項の数9(全18頁)

(21)出願番号 特願平4-295194

(22)出願日 平成4年(1992)11月4日

(31)優先権主張番号 810619

(32)優先日 1991年12月19日

(33)優先権主張国 米国(US)

(71)出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州  
アーモンク (番地なし)

(72)発明者 ウィリアム、ピーター、カーニー

アメリカ合衆国カリフォルニア州、サン、  
ノゼ、カレロ、ヒルズ、コート、7153

(74)代理人 弁理士 須宮 孝一 (外5名)

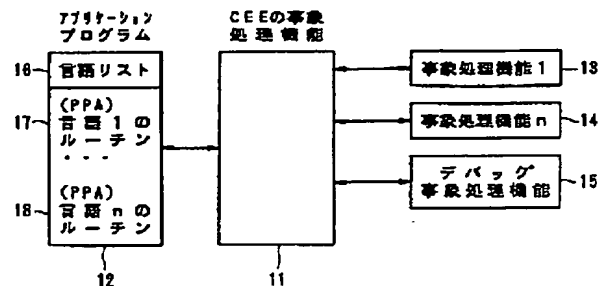
最終頁に続く

(54)【発明の名称】 コンピュータシステムにおいて事象を取り扱う方法及びシステム

(57)【要約】 (修正有)

【目的】 プログラム実行時、関連するパラメータを有する事象を取り扱うための方法と装置の提供。

【構成】 複数のプログラム言語のルーチンを含むプログラムの実行中生じるコンピュータシステム内で事象を取り扱うため、各独自のプログラム言語の数及び識別は言語リストまたは他の等価手段で決定される。独自の事象取り扱い手段(事象処理機能)は各独自のプログラム言語毎に初期化され、プログラム実行時に、イベント処理機能の選択された事象の検出と、そのパラメータが決定される。検出された事象は同報通信または目標に分割され、同時通信事象は(デバッグ事象処理機能を除く)すべての事象処理機能に送られ、目標事象は1つのイベント処理機能に送られる。事象コード及び関連パラメータは、事象処理機能に送られる。この事象処理機能は、事象の成功、失敗または非処理を支持する適当な戻りコードの発生と、選択された事象毎に必要な情報を戻す。



1

## 【特許請求の範囲】

【請求項 1】 関連するパラメータを有しかつ複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中に発生するコンピュータシステム内の事象を取り扱うための方法であって、

プログラムの準備中に使用される独特のコンピュータプログラムを決定するステップと、

独特の各コンピュータプログラミング言語用の独特の事象取り扱い手段であるイベント処理機能を初期化するステップと、

プログラムの実行中選択されたイベントの発生を検出するステップと、

選択されたイベントの関連するパラメータを決定するステップと、

識別事象コード及び関連するパラメータを少なくとも 1 つの事象処理機能に渡すステップとを有するコンピュータシステムにおいて事象を取り扱う方法。

【請求項 2】 共通の実行環境を有するコンピュータシステム内で事象処理機能を操作する方法であって、

(a) 共通の実行環境から事象コード及びパラメータを受け取るステップと、

(b) 事象が事象処理機能によって活動を要求するか及び共通の実行環境に対して“アクションが生じない”ことを応答するかどうかを決定するステップと、

(c) 事象が事象処理機能から情報を必要とし、必要な情報を見つけるために応答するかどうかを決定するステップと、

(d) ステップ (b) または (c) 内で取り扱われない事象を処理するステップと、

(e) ステップ (c) または (d) で実行されたアクションの成功、非処理または障害をステップ (c) で発見された情報とともに共通の実行環境に戻すステップとを有するコンピュータシステムにおいて事象処理機能を操作する方法。

【請求項 3】 複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中発生する事象を取り扱うようになっているコンピュータシステムであって、

プログラムの準備に使用される独特のコンピュータプログラム言語を決定するための手段と、

各独特のコンピュータプログラム言語の独特の事象取り扱い手段である事象処理機能を初期化するための手段と、

選択された事象に関連するパラメータを決定するための手段と、

事象コードと関連するパラメータを少なくとも 1 つの処理機能に通過させるための手段とを有するコンピュータシステム。

【請求項 4】 選択された事象が同報通信タイプかどうかを決定する手段と、

2

すべての同報通信タイプ事象において、初期化されたすべての事象処理機能に関連するパラメータ及び識別事象コードを送る手段とを有する請求項 3 に記載のシステム。

【請求項 5】 プログラムブロッグ領域から取られた高水準言語メンバ識別子を使用することによって事象コード及びパラメータを送る事象処理機能を決定するための手段を有する請求項 3 に記載のシステム。

【請求項 6】 初期化手段はあらかじめ割り当てられたメンバ識別子を使用して各事象処理機能用に割り当てられる名称を形成する手段と、

割り当てられる名称を使用して各事象処理機能の動的ロードを要求する手段とを有する請求項 1 に記載のシステム。

【請求項 7】 事象の処理の成功、非処理または障害を示す事象処理機能から戻りコードを受け取る手段を有する請求項 3 に記載のシステム。

【請求項 8】 DSA の所有権、エントリポイント及びコンパイルユニット識別、ステートメント識別または DSA クラシフィケーションを表す事象処理機能からの情報を受け取る手段を有する請求項 3 に記載のシステム。

【請求項 9】 デバッグ事象処理機能を初期化する手段を有する請求項 3 に記載のシステム。

## 【発明の詳細な説明】

【0001】

【産業上の利用分野】 本発明はコンピュータシステムのプログラム実行管理の分野の方法及び装置に関する。

【0002】

【従来の技術】 コンピュータプログラミング用の種々の高水準言語 (HLLs) (例えば、C、COBOL、FORTRAN、PL/L、etc) は異なる走行時間事象処理の要求がある。1 つの言語で形成される事象は他の言語で定義されない。

【0003】 典型的なプログラム言語で書かれたプログラムの実行は、特定の走行時間サポートコードが存在し、各コンパイラによって生じたコードが正しく作用することができるように初期化される。走行時間サポートはすべてのプログラムがそのプログラムで書かれていない場合に数多く要求される種々の機能を含む。入力/出力操作に対するルーチンは 1 つの例である。

【0004】 いくつかの言語で書かれたルーチンからプログラムをつくることができることが望ましい。従って、走行時間サポートは、言語の任意の要求を異なる要求で同時に満たすことができなくてはならない。多数の実行時間サポートパッケージに現れる共通のルーチンを分離し、それを複数の言語のアプリケーションプログラムをサポートする共通の走行時間支持パッケージに配置し、それによって、言語の特定のサポートコンポーネントのサイズを減少させ及び冗長性を減少させることができる。

50

【0005】共通の実行環境は、「メンバ」と呼ばれる）HLLが種々のリソースを管理することができるバーチャルマシンインタフェースを確立する。このようなバーチャルマシン内において、「事象を無視すること」を含むメンバによって取られるマンドートアクションを取る種々の重要なイベントが生じる。バーチャルマシン内のこのような重要な事象の発生のメンバを通知し、メンバが適当なアクションをとることを可能にする一様な方法が必要になる。さらに、バーチャルマシンは、時間の別々の点でメンバから情報を得て、種々の活動を実行するためにメンバに要求する矛盾のない方法が必要になる。

#### 【0006】

【発明が解決しようとする課題】従来の技術において、事象の取り扱いのためにベースの走行時間及びHLLの特定のコンポーネントの間の異なる特定のインタフェースの類似した知識が必要になる。従来技術によって提供されない必要なものは、異なる高水準言語ルーチンを1つの走行時間環境で交互に作動させることができる、プログラム実行中の事象の取り扱いを管理するための方法及び手段である。

#### 【0007】

【課題を解決するための手段及び作用】本発明は複数のコンピュータプログラム言語で準備されたルーチンを含むプログラムの実行中生じるコンピュータシステム内で事象を取り扱うための方法及び手段を開示する。プログラムの準備において使用される各独自のコンピュータプログラム言語の数及び識別は言語リストまたは他の等価手段を使用して決定され、特定の事象取り扱い手段（事象処理機能）が各独自のコンピュータプログラム言語毎に初期化される。プログラムが実行されるときに、イベント処理機能に関する選択された事象が検出される。選択される事象に関するパラメータは決定される。さらに検出された事象は2つのタイプ、すなわち同報通信または目標に分割される。同時通信事象は（デバッグ事象処理機能を含むまたは含まない）すべての事象処理機能に送られ、それ故、目標事象は1つのイベント処理機能に送られ、事象コード及び関係するパラメータは1つのイベント処理機能に送られる。事象コード及び関連パラメータは、各事象処理機能が、アクションがサポートされるプログラミング言語の文脈の事象に適当ならばなんでも実行するように事象処理機能に送られる。この事象処理機能は、事象の成功、失敗または非処理を支持する適当なリターンコードを発生し、及び選択された事象毎に必要な情報を戻す。本発明は分離した特定のデバッグ事象処理機能を提供する。

#### 【0008】

【実施例】次の用語はこの明細書に使用される。

#### 【0009】CAA

共通のアンカ領域（CAA）はアプリケーションの実行

スレッドの現在の状態を表す制御ブロックである。

#### 【0010】CIB

処理される最後の状態例えば、エラー、例外等に関する情報を含む制御情報ブロック。

#### 【0011】コンディション

この用語コンディション及びエクセプションは、時折相互変換的に使用される。これらのコンディション及びエクセプションは、無効なマシンのアドレスまたはゼロによって割算する試みのようなエラーである。またコンディションは実際のエラーではないが、特定のアテンションを必要とする事象を含む。

#### 【0012】コンディション・マネージャ

はコンディションが生じたときコンピュータシステムの制御を得、種々のシステム及び／または使用者のアプリケーションのルーチンを実行することによってコンディションの取り扱いを管理するプログラムまたは手段である。

#### 【0013】DSA

ダイナミックセイブエリア、例えばスタックフレーム EDB

このEnclave Data Block (EDB) はエンクレープの現在の状態を表し、このエンクレープによって保持されるリソースを表す制御ブロックである。

#### 【0014】エンクレープ

（エンクレープ）Enclaveは1グループの手順の実行をサポートする論理的な走行時間構造である。このエンクレープ内に呼び込む第1の手順は「主な」手順として公知であり、他のものは「サブ」手順として公知である。このエンクレープは割り当てられた記憶装置及び端子を含む高水準言語の意味の範囲を制限する。

#### 【0015】フィードバック コード

フィードバックコードは状態またはエラーの包含（encapsulation）である。

#### 【0016】ヒープ ストレージ

プロセス内で走行するプログラムに関するプログラム記憶領域の発生セグメントの順列化されていないグループ。すなわち、特定のタイプのプログラム記憶領域は発生セグメントの獲得及び配置の論理的な順序がない、またheap内のプログラム記憶領域の発生セグメントが特定されないheapとして見る事ができる。

#### 【0017】イノベーション

イノベーションは論理的な呼び出し／戻り機構によって制御を有するコード「手順」の実行例である。

#### 【0018】OCB

このオプションコントロールブロック（OCB）は走行時間オプションの現在の設定を含む制御ブロックである。

#### 【0019】PCB

このProcess Control Block (P

5

CB)は処理の現在の状態を表し、また処理水準で保持されるリソースを表す制御ブロックである。

#### 【0020】プラットフォーム

プラットフォームは操作システムとプログラムが実行されるコンピュータのハードウェアとの組み合わせである。

#### 【0021】プロセデュア

変化する意味がなく別々にコンパイルされることのできる最も小さいソースシーケンスに対応するコンパイルユニット。

#### 【0022】プロセス

プロセスは特性が実行時間環境によって説明される最も外側の構造である。処理の間のハイクラキカル(hierarchicall)な関係がない。むしろ、他の顕著な論理的に別れた「アドレススペース」によって各処理はいくつかのシステムのリソースにおいて独立して同期的に競争する。処理は1つまたはそれ以上のエンクレーブを有する。

#### 【0023】PPA

プログラム・プロローグ・エリアは標準のコンパイラによって発生し、コンパイルユニットに関する情報を含む。

#### 【0024】スタック・フレーム

スタック・フレームはスタック(stack)の1つのエレメントである。スタック・フレームは、手順が呼び出しのために戻るときに手順を呼び出す及び削除される各時間につくられる。手順呼び出し、実行及び戻りに関するリソースを管理するために使用される。

#### 【0025】スタック・フレーム・ゼロ

スタック・フレーム・ゼロは第1のルーチン用のスタックフレームの直前の概念的なスタックフレームである。第1の手順は、スタック・フレーム・ゼロから呼び出され、終了はスタック・フレーム・ゼロから初期化される。状態の取り扱いの目的でゼロ番目のスタック・フレームは言語用の省略アクションが加えられるフレームである。

#### 【0026】スタティック・ストレージ

このプログラムはエンクレーブ内でプログラムが走行するとき第1回目の新しい割り当てを得、プログラムが戻るときに自動的に再び割り当てられないプログラムの作動記憶装置。スタティックストレージはエンクレーブ内のプログラムの複数の走行にわたって持続する。

#### 【0027】スレッド

スレッドは走行時間環境のプログラムモデル内での実行の基本的ユニットである。それが所有するリソースは機械の状態、スタック及び論理的に明瞭な状態の管理者である。各スレッドは実行毎に同期的に競争する。

【0028】本発明は、種々の高水準言語コンパイラ及び共通の実行環境(CEE)の事象取り扱いユニットに関する特定の走行時間事象取り扱いモジュールの間のイ

6

ンタフェース内で実行される。図1は、本発明に基づいたシステムのブロック構成要素を示す。CEE11の事象取り扱いユニットはアプリケーションプログラム12と事象処理機能13-15との間をインタフェースする。このCEE及び事象処理機能は操作システムから別れたプログラムとして問題の状態で行走するが、必要とされる基本的な変更なしに操作システムの一部として容易に含まれ得る。(単に通常プログラムと称されるアプリケーションプログラムはプログラムの種々の構成要素の準備に使用されるすべてのHLLを識別する言語リスト16を含む。各ルーチン(コンパイルユニット)17-18は書き込まれる言語リストの指示を含む。これはプログラムプロローグエリア(PPA)に割り当てられたメンバ数を配置することによって行うことが望ましい。

【0029】図2は、EHUの主な構成要素を示す。この言語リストプロセッサ21はアプリケーションプログラムの内部の言語リストの内容を見つけこれを解釈するようになっている。この出力は付加ライブラリから必要な事象処理機能を負荷するために事象処理機能イニシャライザ22によって使用されるメンバのリストである。事象処理機能イニシャライザは事象ディテクタ23が初期化が完了した後アクティブになることができるようにする。関連する事象の発生セグメントが事象ディテクタによって検出されたとき、事象処理機能インタフェースユニット24は必要なパラメータを集め、1つまたはそれ以上の事象処理機能に事象のタイプを識別する事象コードとともにそれらを送る。また、事象処理機能インターフェイスユニットは事象処理機能によって供給される戻りコード及び戻り情報を受ける。この事象の検出は公知の従来技術の手段によって実行される。典型的には、この事象はプログラムの活動で明示される。例えば、プログラムが終了するとき、適当な事象処理機能を駆動するCEEを通知する。暗黙の事象はプログラムの実行中に起こるエラーまたは他の状態に関連する。

【0030】定義された事象が起こるときに、正しい事象処理機能にそれを経路指定するために決定の数をつくらなければならない。事象の基本的なタイプは、もし事象処理機能に送られるならば、どのパラメータを送るべきか、及び事象がすべての事象処理機能に関するかどうか、または1つの処理機能のみを呼び出すべきかを決定する。事象が現在実行中のルーチンに対応する事象処理機能にのみ関することが決定されるならば、その事象処理機能の識別はメンバコードを含む現在のルーチンのPPAを検査することによって確かめられる。事象処理機能に順に送られる事象はいわゆる同報通信と呼ばれる。本発明の特定の実施例は、どの事象が同時通信に必要かを必要でないかを決定する。各事象処理機能は、特定の事象が呼び込まれるとき活動をしないオプションを有する。PPAがナンバーコードを含まないならば、目標と

される事象は、事象の所有者を見つけたすまで同時通信である。図3は本発明によるEHUによって実行される主なステップを示す。プログラムに使用される言語の識別はプログラムに含まれる言語リストから決定され（ステップ31）、事象処理機能は各言語毎に初期化される（ステップ32）。関連事象が検出されると（ステップ33）、関連パラメータが決定される（ステップ34）。事象を同時通信するかどうかに関する決定が行われ（ステップ35）、もしそうなら、事象コード及びパラメータは事象処理機能に送られる。各事象処理機能は少なくとも戻りコードに戻り適切に処理される（ステップ36）。非同時通信事象において、正しい事象処理機能の識別は現在実行中のPPAに含まれるメンバ数から決定される（ステップ37）。事象コード及びパラメータは識別された事象処理機能に送られ（ステップ38）、戻り情報は情報処理機能が完了した後に処理される（ステップ39）。

【0031】本発明は、一般化された環境と個々のHLLの特定の環境とを分離し、事象処理機能とデバッグ事象処理機能のそれぞれを介してデバッグを行う。事象処理機能によって提供されるインタフェースは、特定のHLLまたはデバッグの親密な知識を有するためにCEEを必要とすることなく追加の言語及びデバッグがCEEで走行することができるようにする。

【0032】本発明は、バーチャルマシン内でメンバの存在を積極的に識別するための機構を与え、バーチャルマシン内でバーチャルマシンからメンバに通信する手段を確立し、この重要な事象のメンバ通知し、この事象の通信とメンバからの応答のインタフェースを定義し、メンバが、発生する重要な事象のために所望の活動を実行するフレームワークを与え、バーチャルマシンにメンバから情報を提供する機構を提供することによって作動する。

【0033】関係するHLLメンバとしてCEEと適当にインタフェースするためにメンバはロード可能なモジュールの形態で事象処理機能を提供しなければならない。CEEは標準のオペレーティング・システム機能及び実行可能なコードの開始アドレスを使用してロードされる事象処理機能は、オペレーティング・システムによってCEEに戻される。これは種々の重要な事象の発生セグメントと通信し、CEEがメンバから情報を受け取る機構を提供する。特定の部材数は、メンバコードとして1バイトのみが使用されるから、好ましい実施例では256に任意に制限される。

【0034】図4はCEEによって呼び出されるときに事象処理機能によって取られる必要があるアクションのフローチャートを示す。イベントコード及びパラメータはインタフェースユニットから受け取られなければならない（ステップ41）。この最初の決定はこの特定の事象処理機能によってすべてのアクションを必要とするか

に関して行われる。4の戻りコードはアクションが起こらないときに使用される。いくつかの事象がCEEに供給される戻り情報を必要とするから、これらの事象は別に取り扱わなければならない（ステップ45）。残りの事象は特定の言語に適当な従来の技術によって処理される（ステップ46）。成功または失敗の戻りコードは、もしあれば、要求された情報とともにCEEの戻される（ステップ47）。

【0035】対話式デバッグはメンバに関する事象の組から明瞭である1組の事象に関する。重要な事象のこの通信はデバッグイベント・ハンドラとして知られている別の事象処理機能によって行われる。特に、デバッグイベント・ハンドラは関係するデバッグによって提供されなければならない。多数のデバッグが利用可能であるが、関連するデバッグイベント・ハンドラを有する利用可能な唯一の利用可能なデバッグが特定のプログラムの実行によって使用される。

#### 【0036】言語リスト

CEEはプログラム内に収容された言語リストを検査することによって環境の初期化中どのメンバが存在するかを決定する。これは次のステップによって達成される。

【0037】1. 各メンバは固定されたフォーマット名を有する「記号CSECT」を提供しなければならない。

【0038】2. 1組のメンバ記号CSECT名用の弱い外側基準のベクトルが確立される。

【0039】3. メンバがアプリケーション（例えば、ロードモジュール）内にあるならば、標準のリスト編集処理は言語リスト内の割り当てられたスロット内のメンバの記号CSECT用のアドレスを配置するが、メンバがプログラム内にないならばメンバの記号CSECTはインデックスはゼロの値を含む。メンバの存在が検出されるときメンバの事象処理機能はCEEによってバーチャルメモリにダイナミックにロードされる。弱い外側リストを含む言語リスト用のサンプルCSECTは図5に見られる。

#### 【0040】事象処理機能

この事象処理機能は重要なイベントが起こったとき、またはメンバによって保持されたいくつかの情報をCEEが必要とするときにプログラムの実行を通じて何回か呼び出されるメンバ供給ルーチンである。この事象処理機能は、メンバ言語の必要な親密な知識を含むが、CEEは本発明の使用を通じてこのタイプの詳細を知る必要がない。

【0041】CEEの初期化中、CEEはこのアプリケーションに存在するメンバの組を決定する。これらのメンバの存在毎に、CEEは事象処理機能を負荷する。事象処理機能の名称は固定されたプレフィックス及びメンバ部材を結び付けることによって任意につくられる。つけられた名称は、「CEEVxxx」であり、xxx

はメンバ部材である。この名称は、標準の操作システム呼び出しによって事象処理機能を含むモジュールを負荷するために使用される。事象処理機能のアドレスは、後の検索のために保管される。メンバ数は任意であるが、好ましい実施例において、メンバ用の言語リストのロットに対応している。

【0042】IBMシステム/370のリンケージコンベンションを使用して、メンバ事象処理機能へのリンケージは標準のパラメータアドレスリストを含む「BALR 14, 15」及びR1を介して行われる。この第1のパラメータは通常事象処理機能を呼んだ事象のタイプを指示する。追加のパラメータは特定の事象による。

【0043】種々のダンプサービスを処理する間、事象処理機能はダンプ事象コード7とともに呼び出される。ダンプ事象コードはどのダンプサービスを実行するかを

表す関数コードのパラメータを有する。ダンプ事象用の残りのパラメータは特定の関数コードによって変化する。

【0044】この言語ユーティリティ事象6はどの情報を要求するかを説明する関数コードパラメータを有する。ユーティリティ用の残りのパラメータは特定の関数コードによって変化する。

【0045】事象のリスト及びそれらの対応するパラメータを表1に示す。CEEINTはCEEを初期化するCEE内の呼び出し可能な入力点用のシンボルラベルである。CEESTARTは、負荷モジュールを独特に指定し、言語リストのような情報用の負荷モジュールの経路選択を行うためのシンボルラベルである。

【0046】

【表1】



表 1 事象コード及びパラメータ

事象	コード	バーム2	バーム3	バーム4	バーム5	バーム6	同報通信
処理	1	C I B	結果	新しい状態			NO
テーブル 処理	2	C I B	結果	新しい状態			NO
SFOCond 処理	3	C I B	結果	新しい状態	0		NO
オプションProc	4	オプション ブロック	ceestart のアドレス	INPL	512バイト ワークエリア		NO
Mainopts	5	INPL	R13CEEINT へのインバンド	ROCEEINT へのインバンド	R1CEEINT へのインバンド	Main opts	NO
カセスExit	17						YES
言語 ユーティリティ	6	関数 コード					NO
Dumpサービス	7	ファンクション コード					YES
GOTOターゲット DSA	10	ターゲットDSA					NO
DSAXit ルーチン	11	関数ブロック のdsa					NO
Enclave Create	18	プログラムマスク 返却	INPL	メンバー指定スレッド トークンまたは0			YES
Enclave 終了	19	INPL					YES
Process 終了	21						YES
デバッグInfo	16						NO
ATTERM 事象	15						NO
新しい エンクレイブ モジュール	8	新モジュール エントリーポイント	CEESTART または0				NO

表1の最後の列は事象がすべてデバッグ事象機能に対して非同報通信すべきかを指示する。本発明の特定の実施例は、どの事象が同報通信であるかを決定するが全体のエンクレイブ上の効果を有する事象は一般に同報通信である。これは、処理及びエンクレイブの初期化及び終了に関する事象を含む。

【0047】目標の事象（非同報通信）が処理されるときに、関連する事象処理機能が決定されなければならない。これは事象が発生したときに実行されたルーチンにおいてPPAを決定することによって行われることが好ましい。言語を識別する独特の方法が実行される。また、正しい事象処理機能を発見するまでユーティリティ事象を連続して同報通信することが可能である。

【0048】CEEは、プロセス初期化のために及びエ

ンクレイブ初期化のためにメンバー特定初期化ルーチンを呼び出す。このリソース及び能力は2つの事象の間で異なる。次の事象を例として説明する。

40 【0049】・処理初期化

- ・エンクレイブ初期化
- ・処理終了
- ・エンクレイブ終了
- ・走行時間オプション事象
- ・atterm終了事象

CEEは通常の呼び出しコンベンションに適応するために戻りにおいてその元の値を記憶するようになってい

50 【0050】

る。事象処理機能は次のような値戻りコードの1つに対してR15内に戻りコードを設定しなければならない。

13

14

戻り  
コード

意味

4

この事象のためにアクションが起こらない。

16

事象の処理が成功せず、及び/またはプログラムを直  
ちに終了しなければならない。

【0051】CEEは事象処理機能が16の値に戻る  
か、前のリストにない値に戻るならばプログラムを終了  
する。

【0052】処理初期化事象：SHIRI初期化は事象  
17である。この事象は処理水準でHLL部分を持ち出  
すために使用する。事象処理機能呼び出す順序は本発  
明の部分として定義されないが、そうでなければ強制さ  
れる。

【0053】メンバに入力するとき、その呼び出し者の  
レジスタ及びR1を記憶することができるDSAを示す  
事象処理機能R13は、事象コード17の1つのパラメ  
ータを有する標準のO/Sスタイルプリストのアドレス  
を含む。

【0054】事象17及び事象18の組み合わせは与え  
られたアプリケーション用の環境のHLL特定のアスペ  
クトを初期化しなければならない。この事象のためのカ  
ウンタパートは事象21である。

【0055】処理終了事象：この処理終了事象コードは  
21である。この事象は処理水準でHLL部分を終了さ  
せるために使用される。メンバ事象処理機能呼び込む  
順序はこの発明の部分で定義しないが、強制される。

【0056】このプリストは処理の終了用の事象コード  
の1つのパラメータを含むO/Sスタイルのプリストで  
ある。

【0057】この事象はHLLが処理の水準で維持され  
るすべてのリソースを放棄しなければならないことを示  
す。アプリケーションを終了させるためのすべてのHLL  
の意味は事象19エンクレーブ終了事象によってすで  
に達成されていることに留意すべきである。

【0058】この事象用のカウンタパートは事象17で  
ある。

【0059】エンクレーブ初期化事象：このエンクレー  
ブ初期化事象コードは18である。この事象はエンクレー  
ブ水準でHLL部分を初期化するために使用される。  
メンバ事象処理機能呼び出す順序は本発明の一部を形  
成しないが、強制される。すべてのCEEサービスは第  
1の事象の時間に利用可能である。このメンバは以下に  
説明するようにプログラムマスクの要求を第2のパラメ  
ータに配置することによってプログラムマスク設定に影  
響を与える。エンクレーブ初期化事象用のメンバ事象処  
理機能に入力するときに、次のものが利用可能である。

【0060】・R14、15は連係レジスタ

・R12はCAAを指定する。

・R13はDSAを指定する。

・R1は次のリストとともに（パラメータのすべては基

準によって送られる）標準のO/Sスタイルプリストの  
アドレスを含む。

【0061】1. 事象コード18

2. プログラムマスクを最も右に保持するフルワード分  
野、入力時にこの領域はゼロになる。

3. CEEINTへ送られる初期化プリスト（INPL）

4. （CICSのもとで実行中に）メンバ特定スレッド  
トークン、またはゼロ

事象17及び18の組み合わせは与えられたアプリケー  
ション用の環境のHLLの特定の観点を初期化しなけれ  
ばならない。

【0062】エンクレーブ終了事象：このエンクレーブ  
終了事象コードは19である。この事象は、エンクレー  
ブ水準でHLLの部分を終了させるために使用される。

【0063】メンバ事象処理機能に入力中、次のものが  
利用可能である。

・R14、15は連係レジスタ

・R13はDSAを指定する。

・R12はCAAを指定する。

・R1は次のリストとともに（パラメータのすべては基  
準によって送られる）標準のO/Sスタイルプリストの  
アドレスを含む。

—エンクレーブ終了19を指示する事象コード

—CEE初期化中にCEEINTに送られる初期化パラ  
メータリスト

この呼び出しによって、HLLが終了エンクレーブの言  
語意味を補強することによってアプリケーションを意味  
論的に終了させることができる。エンクレーブ関連リソ  
ースは解放されなければならない。この事象は事象18  
のカウンタパートである。

【0064】走行時間オプションイベント：走行時間オ  
プションは数字4である。この事象は能力を制限してい  
る。利用可能なスタックがなく、CEEの呼び出し可能  
なサービスもない。この目的はコンパチブルな態様で走  
行時間のオプションをメンバが取り扱うことができるよ  
うにすることである。このパラメータリストは次のよう  
である。

【0065】1. 事象コード4

2. OCBのアドレス

3. CEESTARTのアドレス

4. 「主な」入力点のアドレス

5. 512バイトのワークエリアのアドレス

アトターム（Atterm）事象：このアトターム事象  
はエンクレーブの終了中に呼び出される。それはすべて

のユーザスタックフレームがスタックから取り除かれた後、エンクレープ終了事象のメンバを呼び出す前に呼び出される。呼び出しを介してCEEのCEEATTRMサービスに登録されたメンバのみが呼び出される。この事象に送られるパラメータリストは1つのパラメータ、事象コード15からなる。

【0066】ダンプ事象処理機能：ダンプ事象処理機能は次のサンプル手順状態に示すパラメータでつくられる。

【0067】一般的なパラメータフォーマット  
手順 (ダンプ 事象 コード 関数 コード [追加 パ  
ーム], f c)

ここでダンプ 事象 コードは7の値を有するフルワード2進の整数である。ファンクション コードは実行されるダンプ関数を特定するフルワード2進整数である。それは次の値の内1つを含む。

【0068】1. なぜそのダンプをとるかについて説明する情報メッセージをダンプする。この関数 コードは、いわゆるCEE3DMPが第1の場所内に取られるダンプの結果のエラーメッセージをプリントする言語ライブラリの出口を特定する。この情報メッセージはエラー毎にMSGFILEに送られるエラーメッセージのコピーである。これらのメッセージはエラーの時にABENDコード、プログラム状態ワード及びレジスタの内容を含む。CEE3DMPがメンバ言語ライブラリによって呼び出されなければ、メンバ言語ライブラリはこの出口でメッセージをプリントしない。

【0069】2. ルーチンのアーギュメントをダンプする。メンバ言語がルーチン毎に利用可能なアーギュメントとローカル変数の間で識別できないならば、それが変数をダンプするダンプサービスによって呼ばれると同じ時間にアーギュメントをダンプしなければならない。

【0070】3. ルーチンの変数のダンプ。これはルーチンによって使用される局所変数と割り当てられた外部変数を含む。メンバ言語ライブラリはそれが決定されるならばこのルーチンによって使用され、または設定された変数のみをダンプしなければならない。

【0071】4. ルーチンに関連するダンプ制御ブロック。これはメンバ言語によってマップされたDSA及びデバッグ用に使用可能なルーチンに関する他の制御ブロックを含む。これは、コンパイラシンボル表及びステートメント表を含む。

【0072】5. ルーチン用の記憶装置のダンプ。これは、自動スタックフレーム記憶装置静的ローカル変数記憶装置を含む。このルーチンと他のルーチンとの間で割り当てられた静的データ記憶装置は、ダンプされなければならない。割り当てられた記憶領域に1つだけのコピーはダンプされなければならない。

【0073】6. スレッドに関するダンプ制御ブロック。スレッドのためのCAAはCEEによってダンプさ

れる。

【0074】7. スレッドに関するダンプ記憶装置。CEEはスレッドに関するすべてのスタック記憶装置をダンプする。メンバ言語はこの出口を使用するスレッドに関連する他のスタック記憶装置をダンプする。スレッドによって使用されるスタック記憶装置はそれが、たとえそれに関連しなくてもダンプされる。データ記憶装置だけをダンプしなければならない。コード含む記憶装置は可能ならばダンプされるべきではない。

【0075】8. エンクレープに関するダンプ制御ブロック。エンクレープ用のEDBはCEE並びにメンバリストによってダンプされる。メンバの言語はメンバリストから遮断された通信領域をダンプしなければならない。通常アプリケーション負荷モジュールの部分である静的なライブラリ通信領域がある。

【0076】9. エンクレープに関するダンプ記憶装置。メンバ言語はこの出口を使用するエンクレープに関連する他の記憶装置をダンプする。これは通常操作システム記憶装置管理に対する直接的な呼び出しを介して得られる記憶装置を含む。データ記憶装置のみをダンプしなければならない。コードを含む記憶装置は可能ならばダンプするべきではない。

【0077】10. ファイルのダンプステータス及び属性。CEEはメッセージサービスによって使用されるファイルの状態及び属性をダンプする。メンバ言語はそれら自身の状態及び属性をダンプしなければならない。これは、アプリケーションを実行する途中で現在のオープンファイル並びに前のオープンファイルを含む。

【0078】11. ファイルに関するダンプ制御ブロック。ファイルステータスを保持する制御ブロック及び他の言語特定制御ブロックがダンプされる。

【0079】12. ファイルに関するダンプ記憶装置バッファ。これらのバッファは動作システムによって割り当てられ、典型的にはCEEヒープサービスを使用しない。CEEヒープサービスによって割り当てられたバッファ記憶装置がダンプされる。

【0080】13. この処理に関するダンプ制御ブロック。処理用のPCBがCEEによってダンプされる。

【0081】14. 処理に関するダンプ記憶装置。データ記憶装置のみがダンプされなければならない。コードを含む記憶装置はダンプされるべきではない。

【0082】15. 追加のグローバルな情報をダンプする。この情報は、ダンプレポートの最後に表れる。ロードされたライブラリモジュールのリストは、追加のグローバルな情報の一例である。

【0083】16. エンクレープの変数をダンプ。これはこのエンクレープによって使用されるすべての静的な外部変数を含む。

【0084】17. ダンプ呼び出しの最後。これはダンプの瞬間毎に事象処理機能に対する追加の呼び出しがな

いことを指示する。

【0085】追加の パームはある関数コードに対する特定のパラメータである。次のダイアグラムは各関数コードを示す。ダンプ 事象 コード7が常に関数コードを越えることに留意すべきである。

【0086】関数コードによるパラメータフォーマット  
手順 (7, 1, fc)

手順 (7, 2 dsaptr, cibptr, caaptr, cdbptr, fc)

手順 (7, 3 dsaptr, cibptr, caaptr, cdbptr, fc) 10

手順 (7, 4 dsaptr, cibptr, caaptr, cdbptr, fc)

手順 (7, 5 dsaptr, cibptr, caaptr, cdbptr, fc)

手順 (7, 6 caaptr edbptr, fc)

手順 (7, 7 caaptr edbptr, fc)

手順 (7, 8 edbptr, fc)

手順 (7, 9 edbptr, fc)

手順 (7, 10 edbptr, fc)

手順 (7, 11 edbptr, fc)

手順 (7, 12 edbptr, fc)

手順 (7, 13 pcbptr, fc)

手順 (7, 14 pcbptr, fc)

手順 (7, 15 edbptr, fc)

手順 (7, 16 edbptr, fc)

手順 (7, 17, fc)

dsaptrはDSAのアドレスを含むフルワードの2進整数である。

【0087】このパラメータはルーチンがCIBを有さないならばゼロである。 30

cibptrはルーチン用のCIBのアドレスを含むフルワードの2進整数である。

caaptrはCAAのアドレスを含むフルワードの2進整数である。

edbptrはEDBのアドレスを含むフルワードの2進整数である。

pcbptrはPCBのアドレスを含むフルワードの2進整数である。

fc (出力)

基準によって送られる12バイトのフィードバック。この出口から3つの状態、実行の成功、ダンプファイルにメッセージを書き込む場合のエラー、またはメンバ言語ダンプ出口が不成功であったことが起こる。

【0088】事象処理機能ユーティリティ事象：例外ハンドリングを含む種々のCEEサービスは、言語特定関数を実行する。これらを実行するために、CEEはメンバ言語ユーティリティ出口を介して情報を受け取る。このユーティリティ出口はCEEを通過させ必要な処理を実行するために必要な情報である。それは、事象コード

6を使用したメンバ事象処理機能の一部である。これらの出口の各々において事象処理機能への説明及び関係を以下に示す。すべての関係は6の事象コードを有し、次に特別の機能コードが続き、ユーティリティに対して特定のパラメータが続く。

【0089】DSA所有権

この出口のために、メンバ言語は、DSAがそれに書かれ、またはそれによって所有されたルーチンに関連するかどうかを特定する。この出口はPPAスタイルのエントリを有しないコードの所有者を決定するためにCEEによって使用される。第1にCEEはこのコードがPPAスタイルの出口を含むかどうかを見るためにチェックする。呼び出し者のDSAの保管されたレジスタのアイキャッチャはそれがCEEエントリ点を示すかどうかを決定するためにチェックされる。これが真実でないならば、CEEは言語が所有権を宣言するまでDSAの所有権のためにメンバ言語出口を呼ぶ。

【0090】パラメータフォーマット

手順 (6, 1 dsaptr)

20 ここで: dsaptr (入力) は活動的なDSAまたはセーブ領域のためにフルワードのポインタである。所有権 (出力) は次のものを含むフルワードの2進の整数である。

0 DSAに対応するソースコードはメンバ言語の中にはない。

1 DSAに対応するソースコードはメンバ言語の中にある。

入力点及びコンパイラユニット識別

この出口のために、メンバ言語は、入力点名、入力アドレス、コンパイラユニット名、コンパイラユニットアドレス及びルーチン用現在の指示、ルーチンと関連する与えられたDSA及びCIBを識別する。この出口はルーチンがPPAスタイルの入力を有しないときのみ呼ばれる。

【0091】パラメータフォーマット

手順 (6, 2 dsaptr, eibptr, コンパイラ ユニット ネーム, コンパイラ ユニット アドレス, エントリ ネーム, エントリ アドレス, コール インストラクション アドレス)

40 ここで, dsaptr (入力) は活動的なDSAまたはセーブエリアに対するフルワードポインタである。

cibptr (入力)

は1つが存在すれば現在の状態用にCIBに対してフルワードポインタであり、このパラメータはゼロである。

【0092】コンパイラ ユニット ネーム (出力)

はDSAに関連するルーチンを含むコンパイルユニットの名前を含むために任意の長さの固定長文字ストリングである。コンパイルユニットの名前を決定することができないならば、このパラメータはすべてのブランクに設定すべきである。このコンパイルユニットの名前が供

給されたストリング内に嵌合することができなければ、それは切り捨てられるべきである。

【0093】コンパイル ユニット 名称 長さ (出力)

は入力時にコンパイラユニット名称の長さ及び出口にストリング内に配置されたコンパイラユニット名称の長さを含むフルワード2進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは256バイトである。

【0094】コンパイラ ユニット アドレス (出力)

はコンパイラユニットの始めのアドレスを含むフルワード2進整数である。このコンパイラユニットのアドレスを決定することができないならば、このパラメータはゼロに設定されなければならない。

【0095】エントリ 名称 (出力)

はDSAに関するルーチンにエントリ点の名称を含むための任意の長さの固定長文字ストリングである。エントリ点名称を決定することができないときは、このパラメータはすべてブランクに設定されなければならない。エントリ点名称が供給されたストリング内に嵌合することができないときは、切り捨てられるべきである。

【0096】エントリ 名称 長さ (出力)

は入口にエントリ点の長さ及び出口でストリング内に配置されたエントリ点の実際の長さを含むフルワード2進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは256バイトである。

【0097】エントリ アドレス (出力)

はエントリ点のアドレスを含むフルワードの2進整数である。エントリアドレスを決定することができないならば、このパラメータはゼロに設定すべきである。

【0098】呼び出し 命令 アドレス (出力)

はルーチンの外側に制御を転送する指示のアドレスを含むフルワードの2進整数である。これは制御がプログラムの中断によって転送されるならば、BALRまたはBASRMのような指示を呼び出し指示のアドレスか、または中断指示のアドレスである。このアドレスを決定することができないならば、このパラメータはゼロに設定されなければならない。

【0099】ステートメント識別

この出口のために、メンバ言語はルーチンへのステートメントナンバが与えられた指示アドレスとエントリアドレスを識別する。また、ルーチン用のDSAのアドレス及びルーチン用のCIBのアドレスが送られ、この場合、現在のレジスタの内容がステートメントナンバを決定するために必要とされる。

【0100】パラメータフォーマット

手順 (6, 3 エントリ アドレス, 呼び出し 指示 ア

ドレス, dsaptr, cibptr, ステートメント id, ステートメント id 長さ)

ここで、

エントリ アドレス (入力)

はルーチンへの入力点のアドレスを含むフルワードの2進整数である。

呼び出し 指示 アドレス (入力)

は識別されるステートメントの指示のアドレスを含むフルワードの2進整数である。

10 【0101】これは、それ自身DSA (例えばフェッチグルーコード) を有しない小さいルーチン内の指示のアドレスであることに留意すべきである。このような場合、小さいルーチンは、ルーチンと呼ばれるステートメント用のコードの延長として考慮される。このような場合、メンバ言語は小さいルーチンの呼び出し者のステートメントの数を送り返さなければならない。

【0102】dsaptr (入力)

は、ルーチン用のDSAのアドレスを含むフルワードのポインタである。cibptr (入力) は現在の状態のCIBのアドレスを含むフルワードのポインタである。CIBがない場合には、このパラメータはゼロである。

【0103】ステートメント id (出力)

呼び出し 指示 アドレスによってポイントされた指示の状態識別子を含む任意の長さの固定長文字である。このステートメントを決定することができないならば、このパラメータはすべてブランクに設定すべきである。ステートメントidが供給されたストリングの範囲内に嵌合されなければ、切り捨てられなければならない。

【0104】ステートメント id 長さ (出力)

30 は入力時にエントリ点の長さ及び出口にストリング内に配置されたエントリ点の実際の長さを含むフルワード2進整数である。コンパイラユニット名称を決定することができないならば、このパラメータはゼロに設定される。ストリングが有するこの最大長さは256バイトである。

【0105】DSAクラシフィケーション

この出口において、メンバ言語は、この手順に関連するDSAのタイプを識別する。

【0106】パラメータフォーマット

40 手順 (6, 4 dsaptr, class)

ここで、

saptr (入力)

はDSAまたはセーブエリアのアドレスを含むフルワードポインタである。class (出力) は送られたDSAのクラシフィケーションを指示する基準によって送られた固定バイナリ (31) である。次に示すものは戻ったフルワードのフォーマットである。コンパイルされたコードからライブラリコードを識別するために迅速にチェックすることができ、必要ならば、コンパイル/ライブラリのタイプをメンバが修飾することができる。例え

21

ば、PL/Iが開始ブロック、Onユニット、手順を識別することができる。

【0107】x' a b c d y y z z'

ここでzzはX' FF' のマックス(max)を有するメンバidである。yyはコンパイルされたコードタイプまたはライブラリコードタイプを修飾するためにメンバによって使用される。

dはライブラリコードならば1  
コンパイルコードならば2である。

【0108】戻りの値はDSAの所有者と独自に関連しなければならない。実際に割り当てられた値は任意であり、本発明に影響することなく交換され得る。例えば、X' 00010005はCOBOL、コンパイラライブラリルーチンに振り分けられ、X' 00020005' はCOBOL、コンパイルされたコードに使用される。

【0109】デバッグ事象処理機能：このデバッグ事象処理機能は名称「CEEVDBG」を有するCEEによって負荷可能でなければならない。使用されるデバッグの使用は、LOADに対してCEE用の装置にその名称を表すことによって実行時間で行われる。ロードの障害は、このプログラムの実行中にデバッグが利用できないCEEに表示する。名称CEEVDGBはロードラ

22

イブラリ内で独特である必要があるだけである。異なるロードライブラリで複数のデバッグ事象処理機能がある。プログラムの特定の実行用に1つだけが使用される。

【0110】このデバッグ事象処理機能は次の内1つが発生するときにロードされ初期化される。

【0111】初期コマンドストリングまたはPROMPTが発見され及びTEST走行時間オプションは有効である。

【0112】エラー状態は第1の時間用に上昇し、TEST走行時間オプションは特定されたエラーサブオプションとともに有効である。

【0113】状態は第1の時間において上昇し、TEST走行時間オプションは特定されたALLサブオプションとともに有効である。

【0114】CEETESTに対する呼び出しはTEST走行時間オプション設定に関係なく行われる。

【0115】CEEはデバッグ事象処理機能を介して事象のデバッグを通告する。このパラメータリストは表2に定義される。

【0116】

【表2】

23  
表2 デバッガ/CEE事象処理機能インターフェイス

デバッガ事象	デバッガコード	パーム 2	パーム 3	パーム 4
上昇した状態	101	CIB	結果コード	
goto	111	d s a		
エンブレフinit	118	クリエ-タの:db		
エンブレフターム	119			
デバッガターム	121			
スレッドint	120	クリエ-タの:db		
スレッドターム	122			
外部エントリ	123	++d s a	cmdストリング	INPL
モジュールロード	124	d s a	モジュールデスクリプタ	
モジュールリート	125	d s a	モジュールネーム	
記憶装置フリー	126	記憶装置	記憶装置長さ	
状態プロモート	127	CIB	結果のコード	
状態goto	128	d s a		
アテンション	129			
デバッガプログラムチェック	130	結果コード		
メッセージダイレクト	131	msg_テキスト	ddname	
CALL CEETEST	132	++d s a	cmdストリング	

表2に使用された用語毎に、次の定義が加えられる。

モジュール ネーム

削除されたモジュールネームのハーフワード・プレフィクスト・ストリングモジュール・デスクリプタはロードされたモジュールを説明する構造。その構造は次のようである。

【0117】dcl 1 モジュール デスクリプタ

3 ロードポイントポインタ

3 固定されたモジュールサイズ

3 エントリポイントポインタ

3 モジュールネームchar (255)

結果コード

生じる状態管理者用の固定された(31)2進値。

【0118】記憶装置長さ

記憶装置のバイト数を含む固定された(31)2進値。

【0119】cmdストリング

デバッガ命令を含むハーフワードプレフィクスストリング。

msg text

CEEメッセージサービスを通して伝達されたテキストのハーフワードプレフィックスドストリング。

ddname

目標ddnameのブランクを有するCL8ストリング、左ジャスティファイ、パッド右を有する。

INPL

40 CEEINTに送られる初期化パラメータリスト。

ノート

1. すべてのパラメータが基準によって送られる。

2. ++hllライブラリルーチンdsaを意味する要求者のdsaはCEEサービスまたは使用者のdsaの要求者である。

3. リターンコードはレジスタ15内に配置される。

・00-成功

・16-デバッガ内のクリティカルなエラー。再び呼び込まない。

50 【0120】先の明細書を使用することによって、本発

25

明は、標準的なプログラミング及び／またはエンジニアリング技術を使用した本発明が実行される。この結果のプログラムはディスク、ディスケット、メモリカード、ROMまたは他のメモリ装置に記憶される。実行のために、このプログラムはコンピュータのRAMにコピーされなければならない。コンピュータサイエンスの技術の分野における当業者は、本発明を実行するために適当な一般的な目的または特定の目的のコンピュータハードウェアとソフトウェアとを容易に組み合わせることができる。本発明の好ましい実施例を詳細に説明したが、この実施例に対する変形例及び適用が特許請求の範囲で述べた本発明の観点から離れずに行われることは明らかである。

#### 【0121】

【発明の効果】以上説明したように本発明によれば、異なる高水準言語ルーチンを1つの走行時間環境で交互に作動させることができる。

#### 【図面の簡単な説明】

【図1】本発明によるアプリケーションプログラム、共

26

通の実行環境（CEE）の事象取り扱いユニット及び事象処理機能の間の関係を示すブロック図。

【図2】本発明によるCEEの事象取り扱いユニットの主な構成要素のブロック図。

【図3】本発明による事象取り扱いユニットによって実行される主なステップによるダイアグラム。

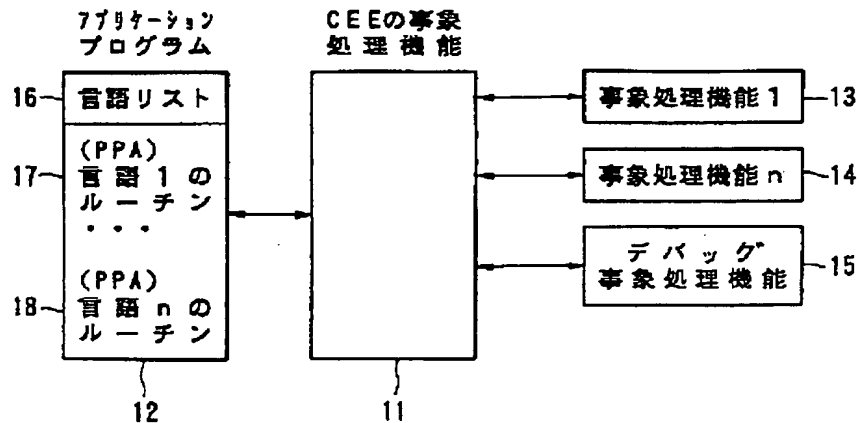
【図4】本発明による事象処理機能のオペレーションにおける主なステップによるダイアグラム。

【図5】本発明によって使用される言語リストをつくるために使用されるアセンブラ言語ソースコードのサンプル説明図。

#### 【符号の説明】

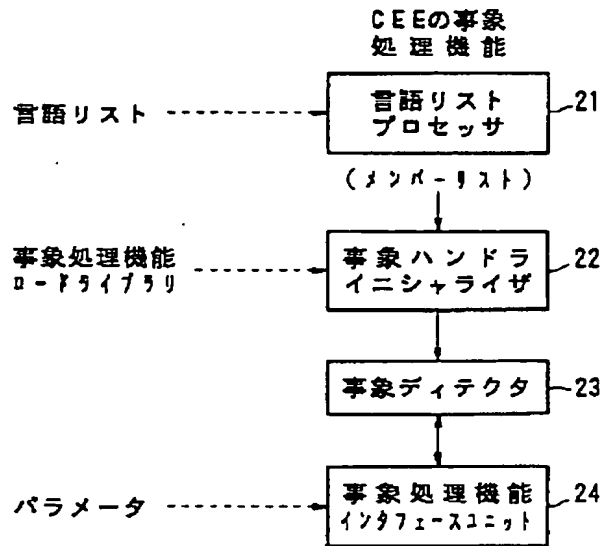
- 11 CEE
- 12 アプリケーションプログラム
- 13-15 事象処理機能
- 17-18 ルーチン
- 21 プロセッサ
- 22 事象処理機能イニシャライザ
- 23 事象検出器

【図1】

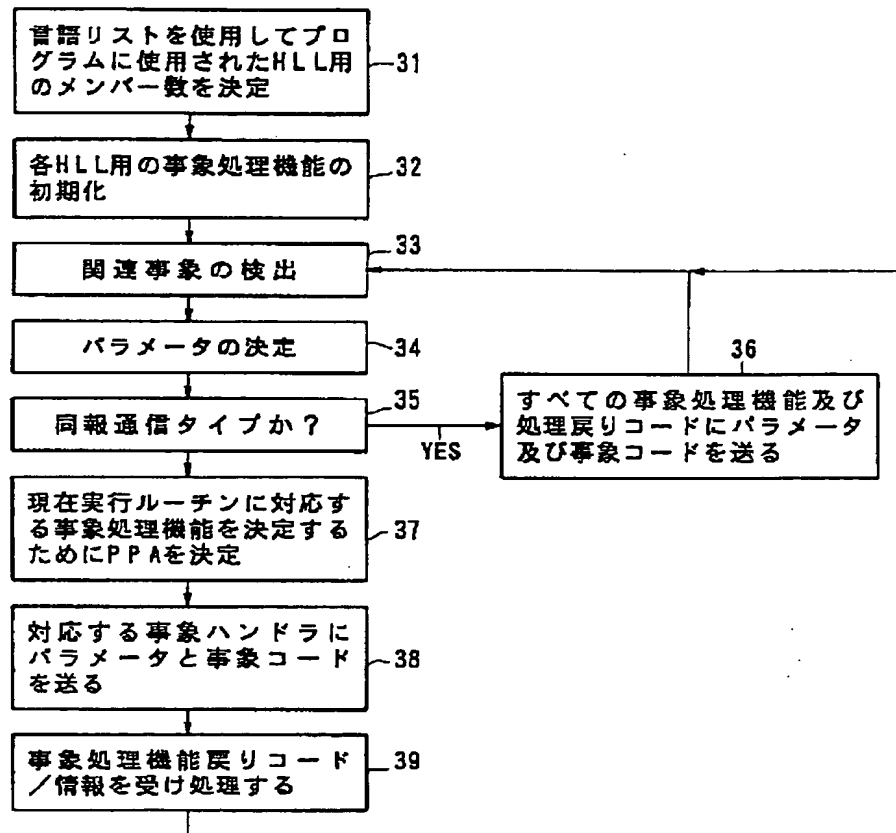




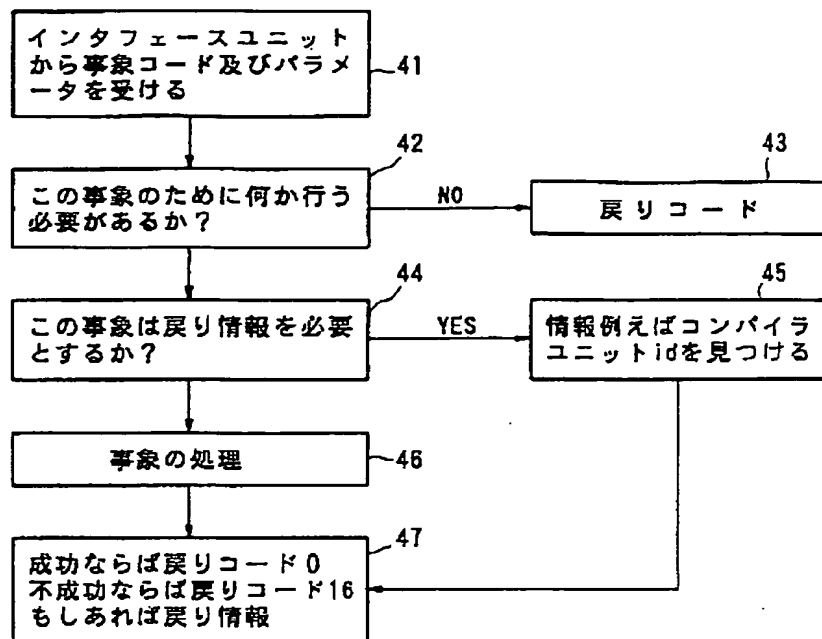
【図2】



【図3】



【図4】



【図5】

```

CEECELLST CSECT ,
CEECELLST RMODE ANY
CEECELLST AMODE ANY
...
* 実際のリストの開始
ENTRY CEECELLST
CEECELLST DS    00          CEE 言語リスト
WXTRN CEESEG000
DC    A(CEESEG000) 00 振り分けられず
WXTRN CEESEG001
DC    A(CEESEG001) 01 コモンサービス
WXTRN CEESEG002
DC    A(CEESEG002) 02 コンパイラ-x
WXTRN CEESEG003
DC    A(CEESEG003) 03 コンパイラ-y
WXTRN CEESEG004
DC    A(CEESEG004) 04 プログラム-x
WXTRN CEESEG005
DC    A(CEESEG005) 05 コンパイラ-z
WXTRN CEESEG006
DC    A(CEESEG006) 06 コンパイラ-j
WXTRN CEESEG007
DC    A(CEESEG007) 07 コンパイラ-k
WXTRN CEESEG008
DC    A(CEESEG008) 08 コンパイラ-l
WXTRN CEESEG009
DC    A(CEESEG009) 09 コンパイラ-m
WXTRN CEESEG010
DC    A(CEESEG010) 10 コンパイラ-n
WXTRN CEESEG011
DC    A(CEESEG011) 11 デバッカ
WXTRN CEESEG012
DC    A(CEESEG012) 12 コンパイラ-o
WXTRN CEESEG013
DC    A(CEESEG013) 13 コンパイラ-p
WXTRN CEESEG014
DC    A(CEESEG014) 14 コンパイラ-q
WXTRN CEESEG015
DC    A(CEESEG015) 15 アセンブラ
WXTRN CEESEG016
DC    A(CEESEG016) 16 コンパイラ-r
DC    A(0)          ダミーはX'00'を含まなくてはならない
DS    00
LLISTEND DC A(0)      リストの端部のマーク
END

```

フロントページの続き

(72) 発明者 ローレンス、エドワード、イングランド  
 アメリカ合衆国カリフォルニア州、モーガ  
 ン、ヒル、ラ、カナダ、コート、520  
 (72) 発明者 ゲーリー、ジョン、ホクマス  
 アメリカ合衆国カリフォルニア州、サン、  
 ノゼ、モーゼル、ドライブ、6787

(72) 発明者 ブライアン、オーウィングス  
 アメリカ合衆国カリフォルニア州、サン、  
 ノゼ、マカティ、サークル、5703、アパー  
 トメント、エイチ

(72)発明者 エリック、リン、ポーター  
アメリカ合衆国カリフォルニア州、フリー  
モント、ノース、モレイ、ストリート、  
43862

(72)発明者 アルフレッド、ウィリアム、シャノン  
アメリカ合衆国カリフォルニア州、モーガ  
ン、ヒル、ラ、クロス、ドライブ、830

(72)発明者 ロバート、アーロン、ウィルソン  
アメリカ合衆国カリフォルニア州、サン、  
ノゼ、チェルトナム、ウェイ、51